Configuration Manual

MSc Research Project
Data Analytics

Ronu Skariah
Student ID: x21159840

School of Computing
National College of Ireland

Supervisor: Noel Cosgrave

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Ronu Skariah
Student ID:	x21159840
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Noel Cosgrave
Submission Due Date:	15/12/2022
Project Title:	Configuration Manual
Word Count:	636
Page Count:	7

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	15th December 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Ronu Skariah
x21159840

1 Introduction

This configuration manual will provide a an in-depth overview of all the software and hardware packages used in the implementation of the research. This will explain the details step by step procedure from the installation required to setup the algorithm execution and evaluation. The below section describe each steps followed.

2 Hardware Configuration

This research implementation was performed on Acer Aspire 5 laptop and the configurations are show in Figure 1

Processor	11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz
Installed RAM	8.00 GB (7.78 GB usable)
System type	64-bit operating system, x64-based processor

Figure 1: System Requirements.

3 GPU Configuration

The implementation has been done using python web IDE, Google Colab. The gpu used in Google Colab Pro subscription is shown in the Figure 2

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   45C    P0    26W /  70W |      0MiB / 15109MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Figure 2: Google Colab Pro gpu Configuration used.

4 Software Configuration

The software requirement for running the Google Colab is show in Figure 5

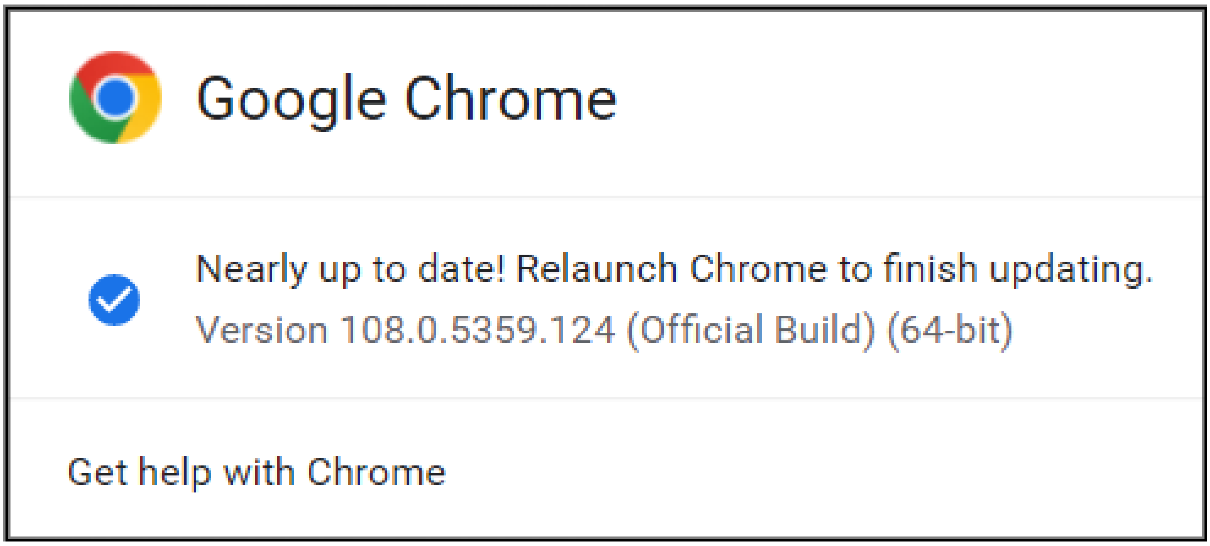

Figure 3: Google chrome version.

5 Package Installations

The Mask RCNN model run on top of TensorFlow 2.2.0 and Keras 2.3.1. The python package required for the complete execution of the implementation is shown in Figure 4

```
# Install the dependencies for the implementation and please restart the runtim
!pip install tensorflow==2.2.0
!pip install keras==2.3.1
!pip install opencv-python
!pip install h5py==2.10.0
!pip install imgaug
!pip install -U scikit-image==0.16.2
!pip install IPython[all]
```

Figure 4: Python package installations.

6 Data Acquisition and Preparation

The research implementation uses two image datasets. The first dataset is a car model detection dataset which has more than 2,000 images of car models and it is obtained from roboflow public repository[1]. The second dataset is the CNR Parkit dataset [2] which is the image dataset that has the collection of more than 1,5000 images of a different parking lot across different climates, this dataset is obtained from a public web repository.

6.1 Data Preparation

The object detection dataset which is the car model detection dataset which has more than 2,000 images are used for training the model and hence they are annotated using an online annotation tool. [3] These images are annotated using a rectangular bounding box selector and the annotations are stored in XML format. The image name and the annotation file name are named identical, Figure 5 show the sample annotation results.

Figure 5: The annotation sample.

[1]https://universe.roboflow.com/mxk/car-model-detection/dataset/
[2]http://cnrpark.it/
[3]https://www.makesense.ai/

The images are partitioned as train, test, and valid based on the data split ratio of 9:1:1. Figure 6 shows the folder structure. The entire folder will be zipped and uploaded to google drive.

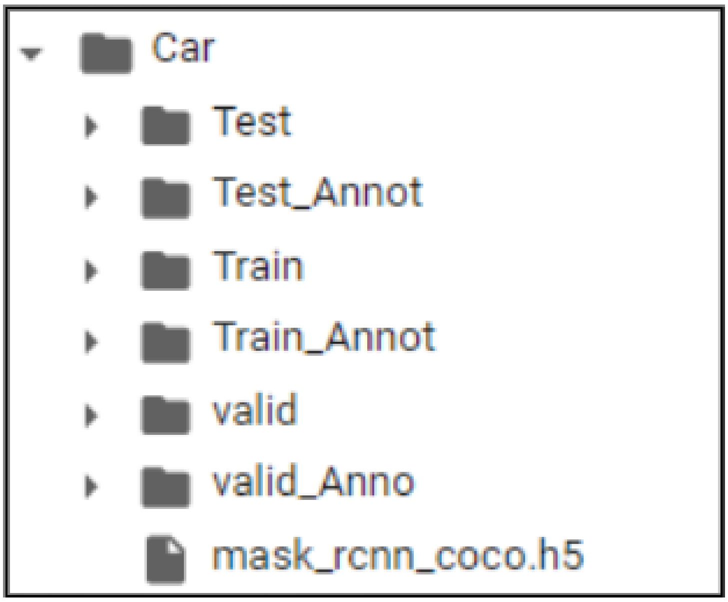

Figure 6: Partition of dataset.

Based on Figure 7 the CNR Parkit dataset is structured and this folder is zipped and uploaded to the same google drive.

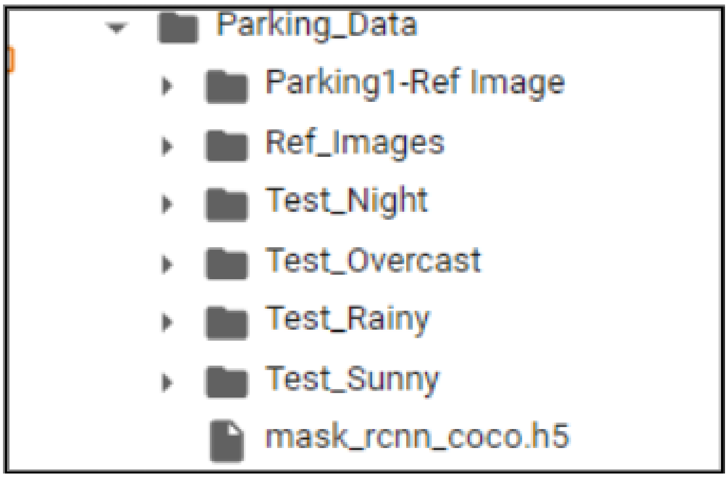

Figure 7: Partition of dataset.

All the datasets will be unzipped from the mounted drive to the Google Colab environment using the shutil package in python as shown in the Figure 9 and 8.

```
[ ] # Mount the google drive to obtain the dataset
    from google.colab import drive
    drive.mount('/content/drive')

    Mounted at /content/drive
```

Figure 8: Mounting google drive.

```
# Unzipping the dataset from drive to colab enviroment
import shutil


# Path of the file
filename = "/content/drive/MyDrive/Car.zip"

# Target directory
extract_dir = "/content/"

# Unzip the file
shutil.unpack_archive(filename, extract_dir)
```

```
#Upload your dataset and place it to /content/Mask_RCNN/
# importing shutil module
import shutil


# Path of the file
filename = "/content/drive/MyDrive/Parking_Data.zip"

# Target directory
extract_dir = "/content/Mask-RCNN-TF2/"

# Unzip the file
shutil.unpack_archive(filename, extract_dir)
```

Figure 9: Partition of dataset.

7 Stage 1: Modeling of Object Detection Model

The Mask RCNN model is used for the detection of cars, which is trained using the car model detection dataset.

7.1 Cloning and Installing the Mask RCNN Model

The Mask RCNN model for TensorFlow 2.2.0 will be cloned from GitHub and installed in the Google Colab environment.Figure 10 shows the cloning of the Mask RCNN model and its installation.

7.2 Training and Evaluation of Mask RCNN Model

The model training will be done on Google Colab gpu and the gpu specification is shown in Figure 2. Training will be performed using the train and validation dataset split from car model detection dataset on a split ratio of 9:1:1. The model was trained for 5 epoch for 131 steps per epoch. The Figure 11 below shows the training progress. The model reference weight is stored in same dataset folder.

The model was evaluated using the test data from the split set using the best trained weight. The mean average precision score which was the mean of the IoU values of all the train and test dataset.

```
[ ]  #Clone the Mask RCNN model for Tensforflow >2.0.0
     !git clone https://github.com/ahmedfgad/Mask-RCNN-TF2.git

     Cloning into 'Mask-RCNN-TF2'...
     remote: Enumerating objects: 1440, done.
     remote: Total 1440 (delta 0), reused 0 (delta 0), pack-reused 1440
     Receiving objects: 100% (1440/1440), 156.99 MiB | 50.44 MiB/s, done.
     Resolving deltas: 100% (796/796), done.

     # Install the Mask RCNN model
     %cd Mask-RCNN-TF2/
     !python setup.py install
```

Figure 10: Cloning and installation of Mask RCNN from github.

```
WARNING:tensorflow:Model failed to serialize as JSON. Ignoring... cannot pickle '_thread.RLock' object
Epoch 1/5
131/131 [==============================] - 7315s 56s/step - loss: 0.3688 - val_loss: 0.2415
Epoch 2/5
131/131 [==============================] - 7132s 54s/step - loss: 0.0884 - val_loss: 0.1755
Epoch 3/5
131/131 [==============================] - 7124s 54s/step - loss: 0.0937 - val_loss: 0.2412
Epoch 4/5
131/131 [==============================] - 7267s 55s/step - loss: 0.0718 - val_loss: 0.1903
Epoch 5/5
131/131 [==============================] - 7114s 54s/step - loss: 0.0667 - val_loss: 0.3055
```

Figure 11: Training progress status.

```
[ ]  # evaluate model on training dataset
     train_mAP, t_precision, t_recall, t_overlap = evaluate_model(
         train_set, model, cfg)
     print("Train mAP: %.3f" % train_mAP)

     Train mAP: 0.905

[ ]  test_mAP, test_precision, test_recall, test_overlap = evaluate_model(
         test_set, model, cfg)
     print("Test mAP: %.3f" % test_mAP)

     Test mAP: 0.919
```

Figure 12: Evaluation results of model.

8 Stage 2: Detection of Empty and Occupied Parking Space

Detection of empty and occupied spaces in a parking lot will be performed using the object detection and intersection over union technique. The Figure 13 shows the results and execution progress of the concept. The trained weights are store in the same dataset folder.

Figure 13: The final results of detection.